**Actividad UF5 UML**

Integrantes actividad:

Vesselin Stanev

José María Marqués González

José ignacio Gutiérrez Cerrato

Daniel Pavón Gómez

URL del repositorio GITHUB compartido: <https://github.com/Vasi81/Calculadora_Entornos>

Nos ponemos de acuerdo para crear una carpeta de trabajo dentro del repositorio llamada UML.

En esta carpeta vamos almacenado los distintos ficheros drawio de la actividad.

**Drawio es la herramienta CASE** que usamos para realizar la actividad.

Se realizan varias reuniones para afrontar la actividad en ambos requerimientos y mediante git vamos realizando distintas versiones en el repositorio.

**Lo primero que hacemos todos los integrantes de la actividad antes de nada es actualizar nuestro repositorio local con el repositorio remoto, de esa manera todas comenzaremos con la misma imagen del proyecto “git pull origin main”.**

Requerimiento 1:

Caso de uso:

Identificamos dos actores principales, el vendedor/empleado y el actor final como cliente.

El actor vendedor puede realizar los siguientes casos de usos:

🡪Venta Artículo

🡪Mantenimiento Clientes

🡪Mantenimiento Artículos

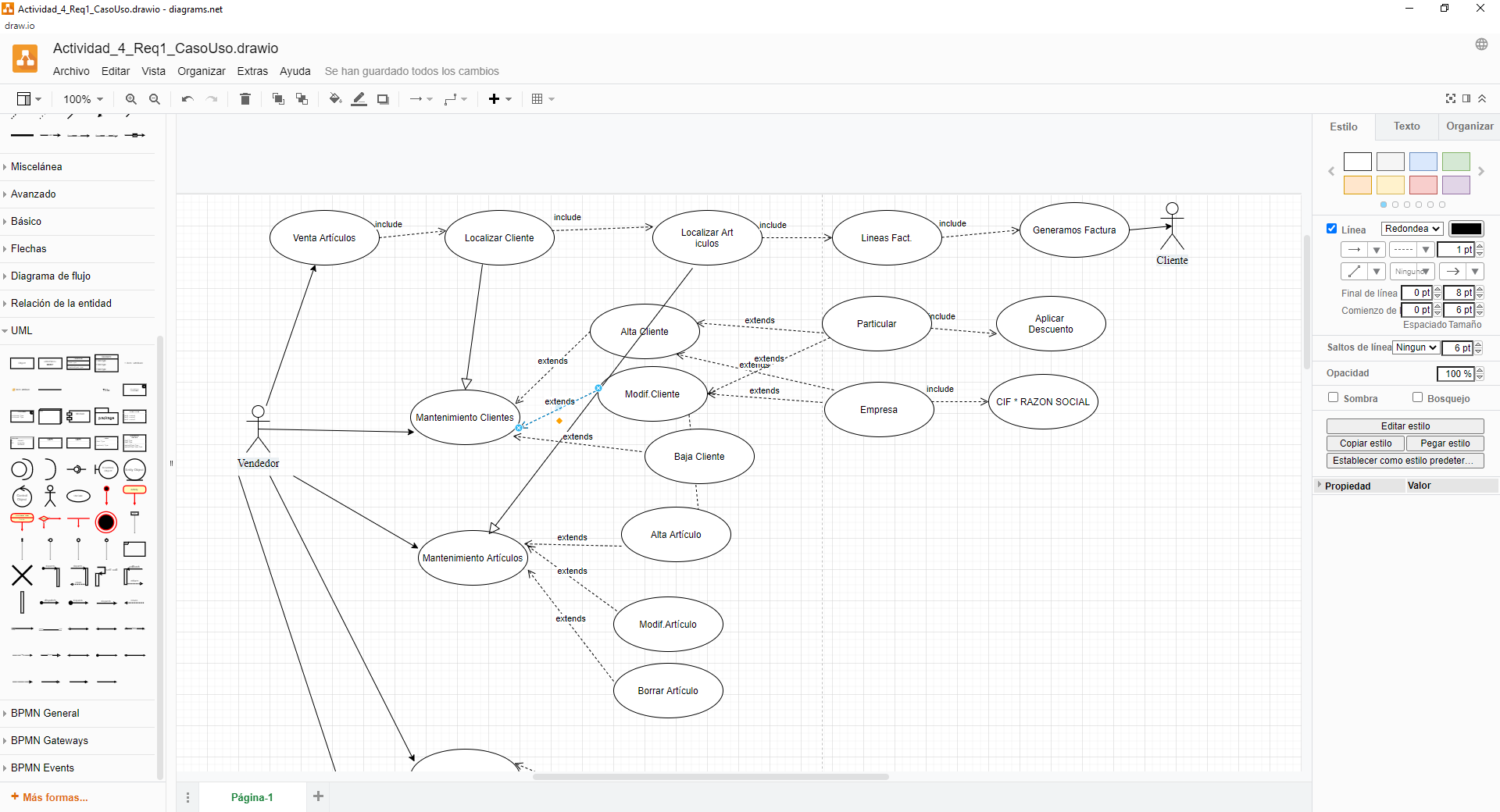
🡪Mantenimiento de facturas

🡪Informes

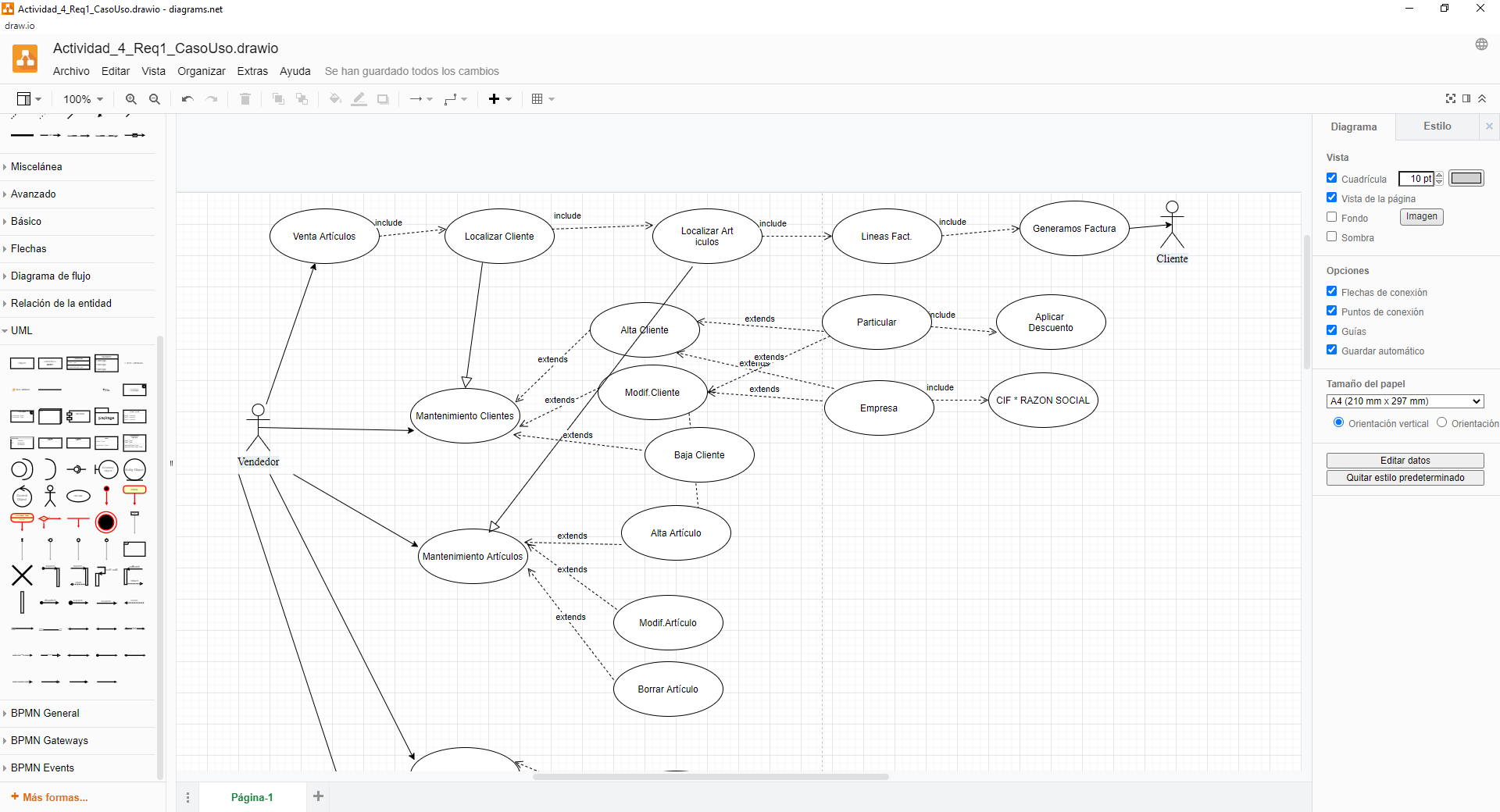
La parte más compleja del proceso es la venta del artículo hasta generar la factura

para el cliente.

El caso de uso “Venta Artículos” desencadena el caso de uso ·Localizar Cliente”, que a su vez hereda del caso de uso “Mantenimiento Cliente” , en el caso de que el cliente no existiera o fuera necesario modificar algún dato que luego será reflejado en la factura.

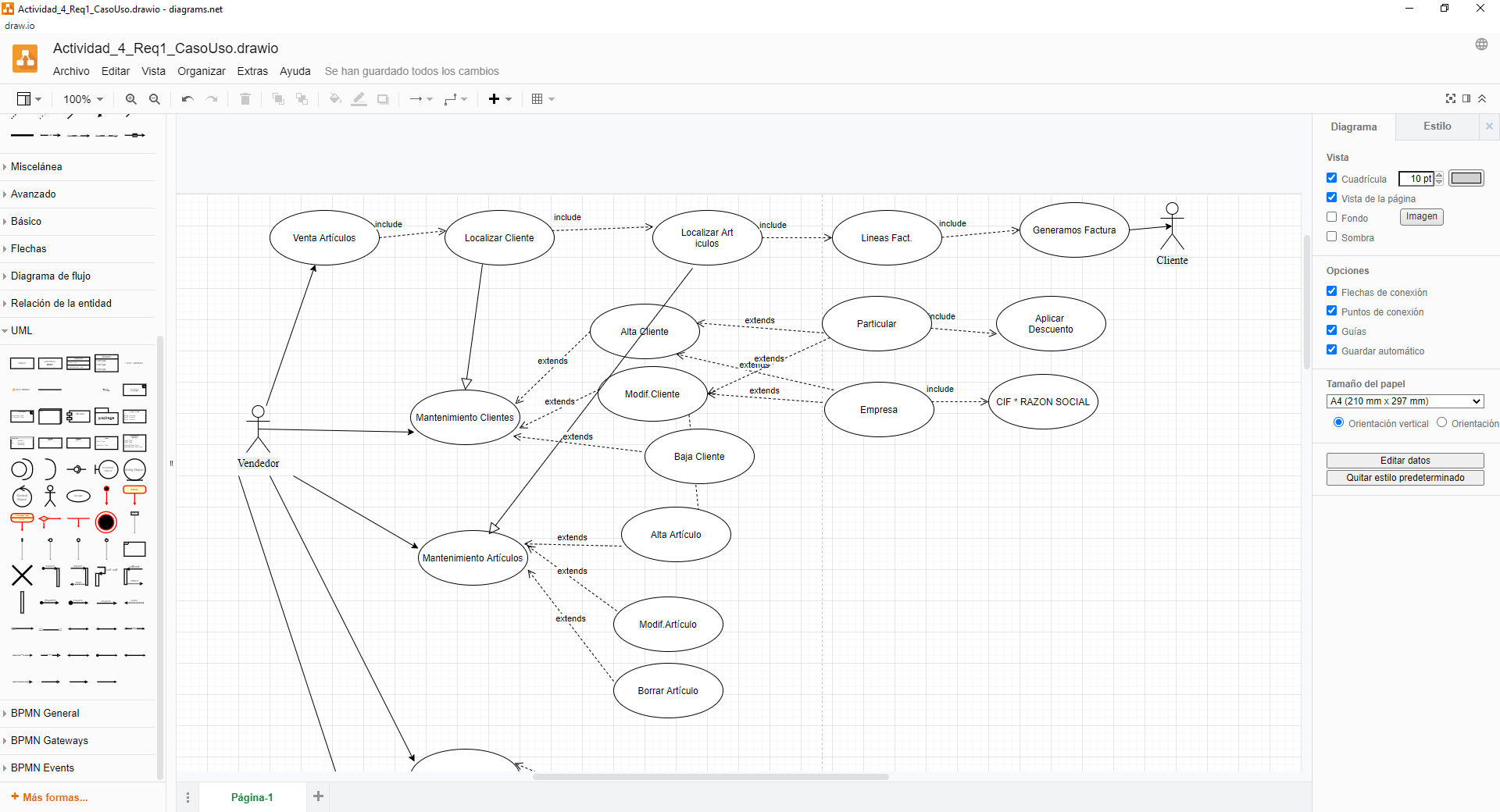


Una vez localizado el cliente para tramitar la factura, localizamos el articulo o artículos que queremos ir incluyendo en la factura que a su vez hereda del caso de uso “Mantenimiento Articulo”.



Una vez finalicemos con las líneas de los distintos artículos introducidos desencadenamos el caso de uso “Generar Factura”, donde deberíamos indicar los datos del Cliente, tanto si es particular (con su descuento) como si es empresa con su CIF y razón social.

Dentro los casos de usos de mantenimiento es caso mas particular sería al que se refiere al cliente, ya que como podemos observar tanto el caso de uso “Alta Cliente” como “Modif. Cliente” pueden desencadenar un caso de uso de tipo extends , ya que puede ir por la opción de cliente o de empresa.



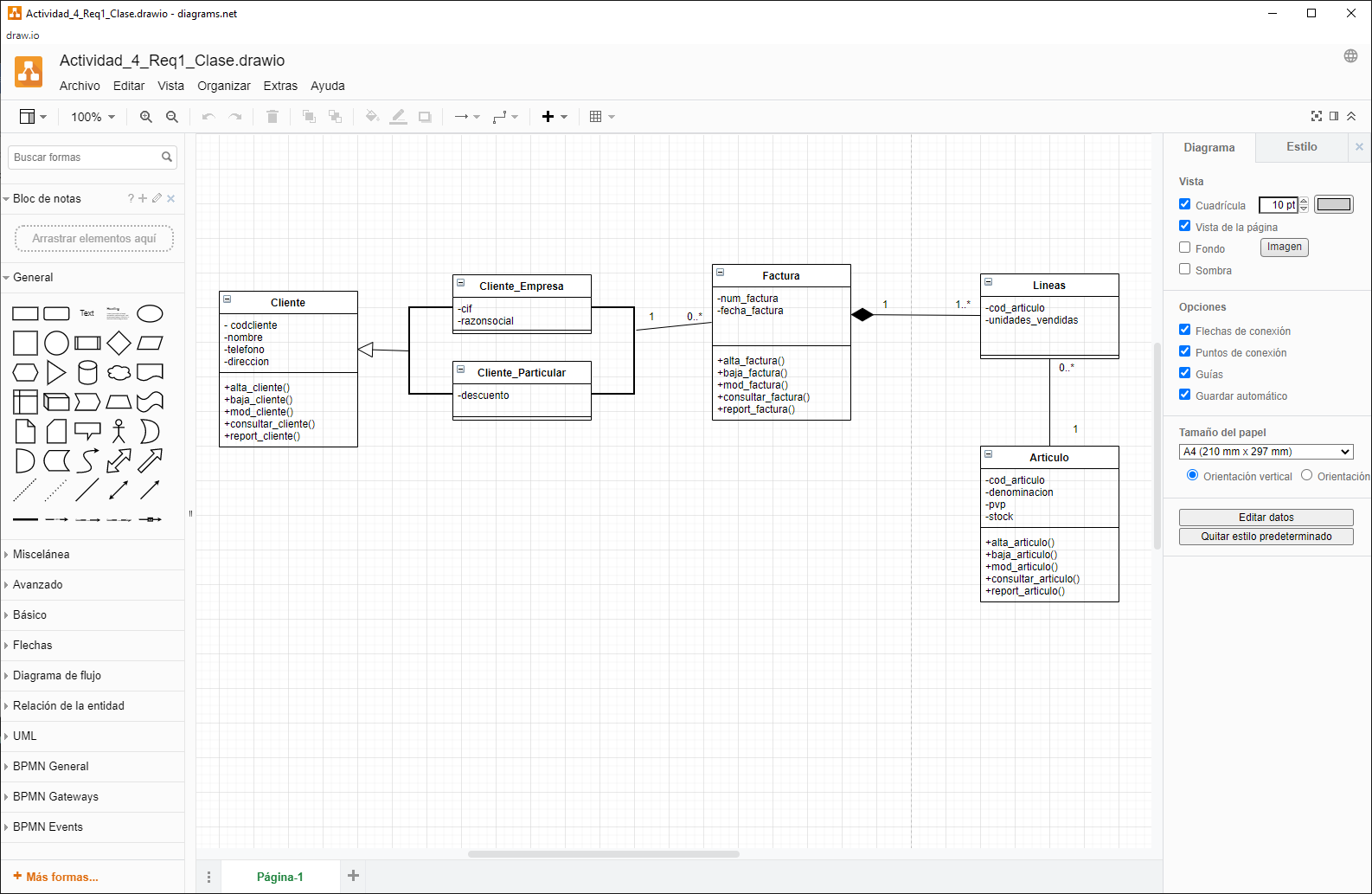
En el resto de casos de usos de Mantenimiento usamos extends para facilitar la elección de las distintas opciones como Alta,Baja,…

En el caso de Informes usamos extends para facilitar la opcionalidad del menú.

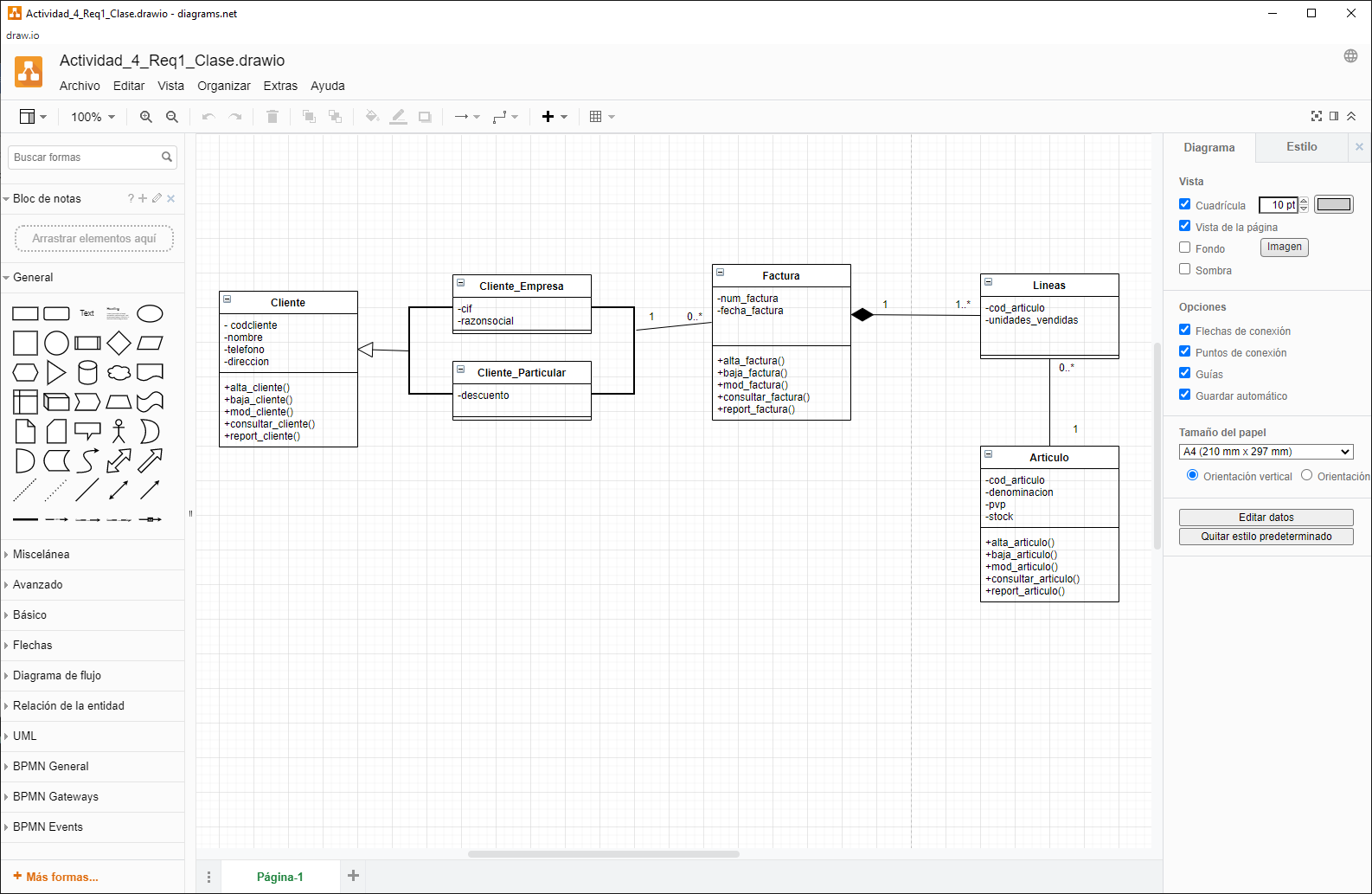
Requerimiento 1:

Diagrama de clases:

Como indicamos en el diagrama de casos de uso, lo primero que reflejamos es la clase cliente, que es heredada por **Cliente\_Empresa y Cliente\_Particular**.

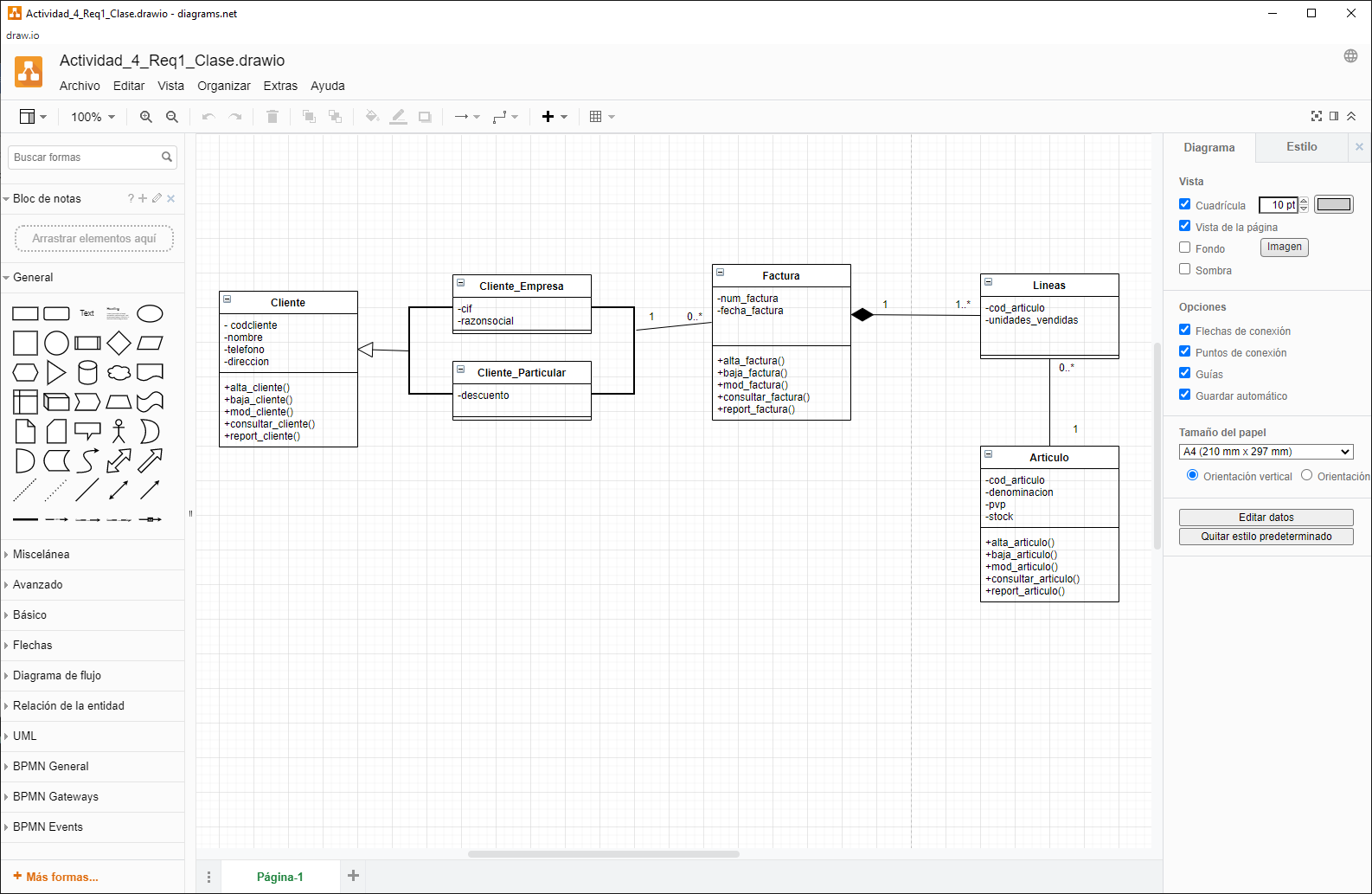


De esas dos clases tenemos una relación por asociación con la clase Factura indicando la cardinalidad 1 – 0.\* , es decir que un cliente (Particular/Empresa) tiene una factura o muchas facturas o ninguna factura y figurar como cliente y una factura tiene un único cliente.



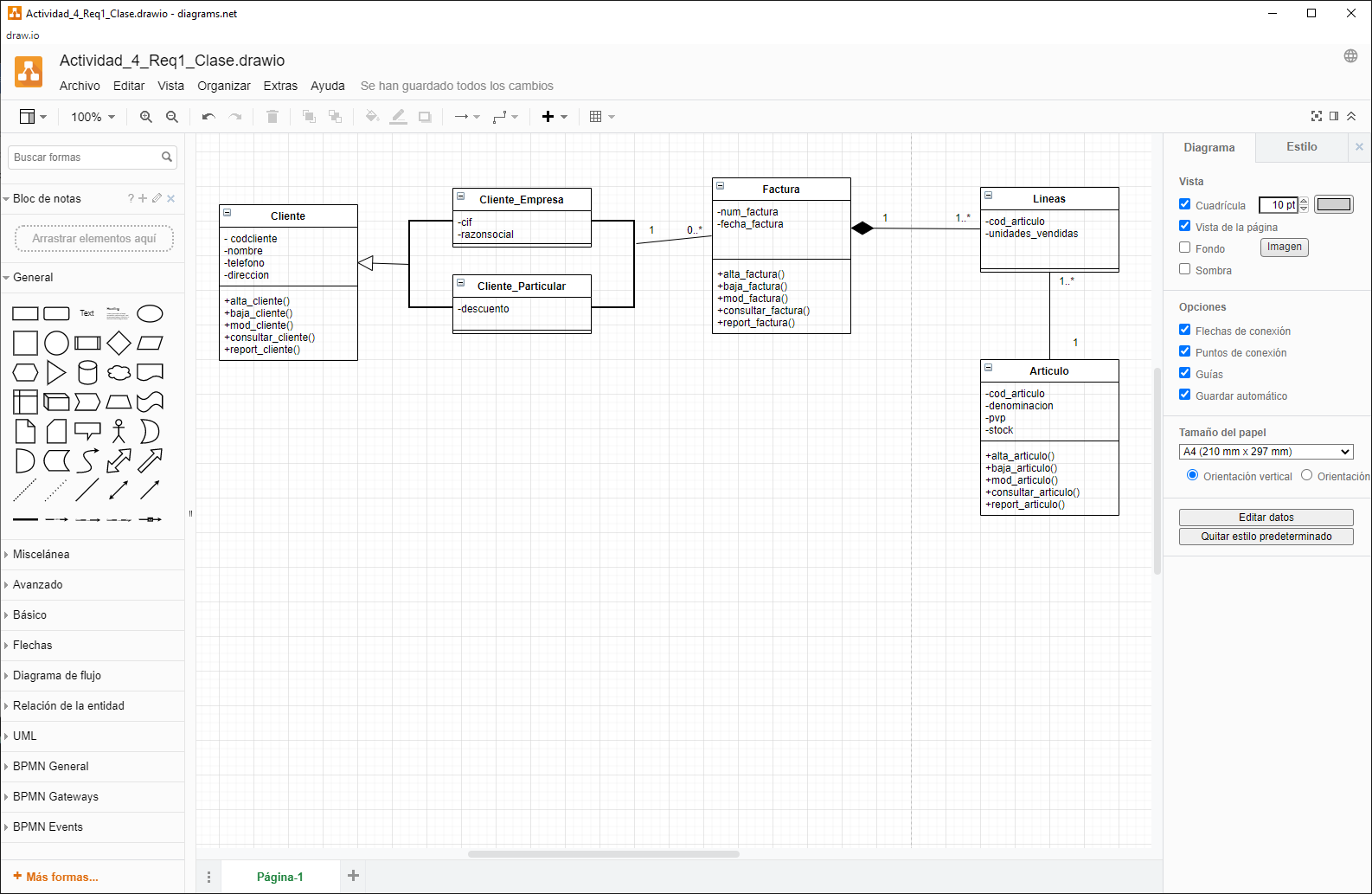
A continuación, una factura tiene distintas líneas sobre el concepto/articulo para facturas, no podemos tener facturas sin concepto/articulo. Esto genera una dependencia de composición entre la factura y las líneas, es decir, si la factura desaparece desaparecen las líneas.

Como podemos observar, de una factura podemos tener 1 o muchas líneas.



Por último, tenemos una relación de asociación entre las líneas y los artículos, ya que una línea tiene uno o muchos artículos.

Con la cardinalidad indicamos que dentro de una línea tenemos un artículo, pero ese artículo puede estar en otras líneas de otras facturas.



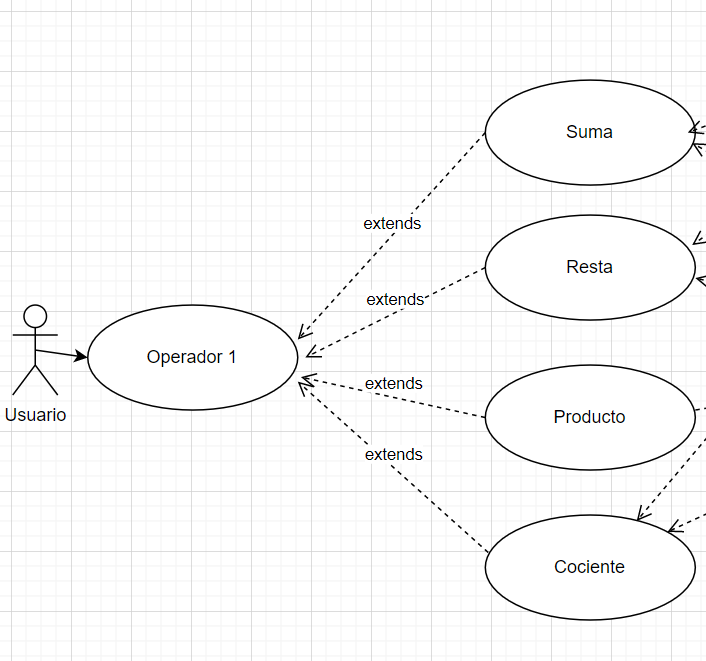
Requerimiento 2:

Caso de uso:

Identificamos un actor principal que es el usuario.

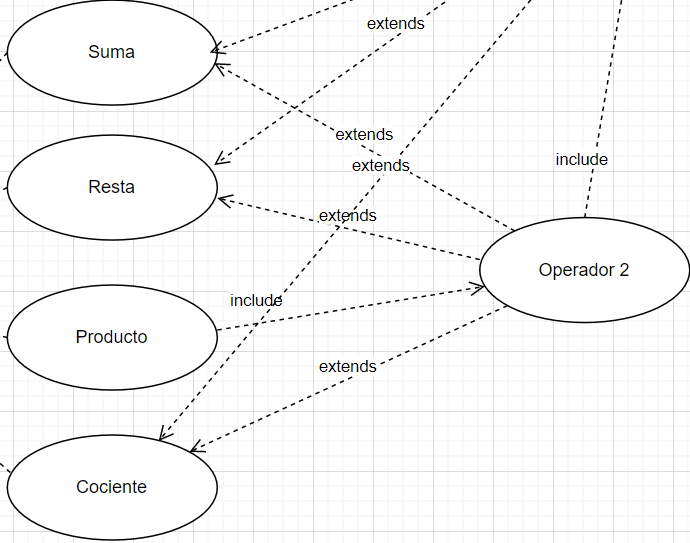
El usuario puede realizar el caso Operador1 que es el inicial, es donde introducimos el primer operador.

El primer caso de uso “Operador1” extiende (extends) los casos “Suma”, “Resta”, “Producto” y “Cociente”. Se usa extends ya que puede realizar cualquiera de ellos.



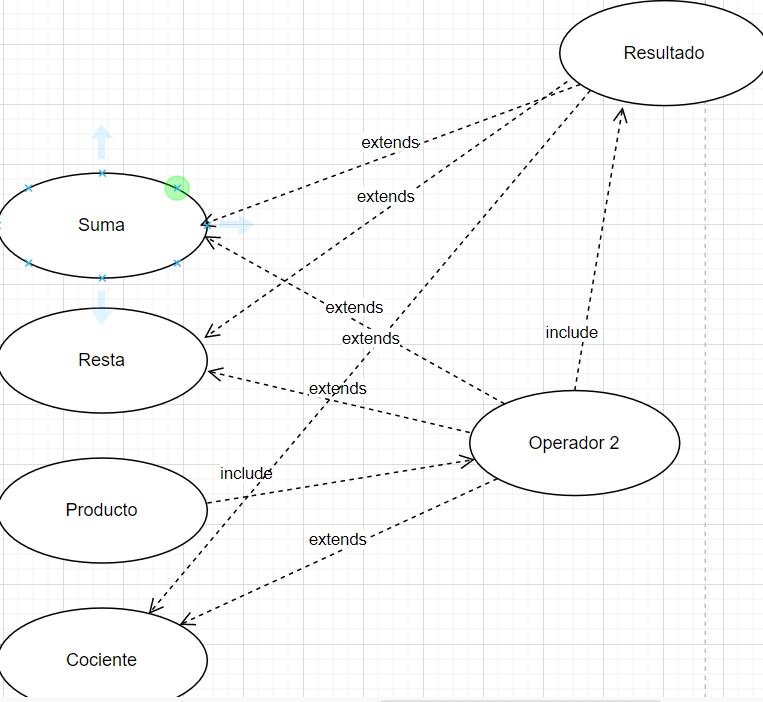
A su vez los casos de uso “Suma”, “Resta” y “Cociente” extienden el caso de uso “Operador2”. Utilizamos extends porque los tres casos pueden realizar el caso “Operador2” o no, ya que existe la posibilidad de que “Suma” y “Resta” usen el valor acumulado y “Cociente” use el método inverso o raíz cuadrada, realizando solamente el caso “Operador1”.

En cambio, para el caso de uso “Producto” utilizamos include ya que tiene que realizar obligatoriamente el caso “Operador2”.



Para el último caso de uso “Resultado” usamos extends en cuanto a “Suma”, “Resta” y “Cociente” ya que podría realizar cualquiera de ellos.

Con el caso “Operador2” usamos include porque tiene que realizar el caso “Resultado”.



Requerimiento 2:

Diagrama de clases:

El diagrama se compone de 5 clases:

* Suma
* Resta
* Producto
* Cociente
* Principal

Las clases de Suma, Resta, Producto y Cociente usan a la clase Principal para crear objetos o para usar sus métodos. Por esa razón hay una relación de dependencia.

